Laboratorul 02.

Observație

Datele de intrare se vor da fie ca parametri în linia de comandă, fie ca valori fixe ale aplicațiilor!

Problema 1

Să se determine numărul de apariții a șirului s_2 în șirul s_1, folosind:

- metoda `substring`;
- metoda `indexOf`.

Exemplu

```
String s1 = "si";
String s = "sir1 si cu sir2 fac un sir3";
```

Rezultat: 4.

```
String substring(int beginIndex, int endIndex);
int pos = s1.indexOf(s2, startingPosition);
```

Problema 2

Să se scrie o metodă care primește ca argumente un șir s_1 și un cuvânt s_2. Metoda întoarce numărul de apariții al cuvantului s_2 în șirul s_1. Se consideră următoarele situații:

1. Se folosește metoda `split` a clasei `String`, iar ca delimitator unic al cuvintelor caracterul spațiul.
2. Se folosește clasa `StringTokenizer`, iar ca separatori se vor considera caracterele `: , . - ?` spațiul și new line.

Exemplul 1: În „sir1 si sir2 sunt 2 siruri” metoda va returna 0 dacă se va căuta „sir”, respectiv 1 dacă se va căuta sir2.

Exemplul 2: În textul „Marius, de ce nu l-ai ajutat pe George? Era de datoria ta sa vezi ce poti face.” cuvântul „ce” apare de 2 ori, iar „l” o singură dată.

Problema 3

Să se genereze un obiect de tip `Vector` ce conține 20 de numere întregi din intervalul $[0..10]$. Acestea se vor genera random, folosind metoda `nextInt`.

1. Să se scrie o metodă care primește ca parametri un obiect de tip `Vector`, generat anterior, şi un număr întreg x. Funcția va elimina fiecare apariție a lui x din vector și va întoarce numărul de apariții al lui x. Testați metoda implementată.
2. Să se scrie o metodă `main` în care, pentru vectorul generat anterior, se determină valoarea componentei maxime, poziţia componentei minime şi media aritmetică a elementelor din vector.

Atentie! Se vor folosi metode din clasa `Vector` pentru operatiile cerute!

```
Random generator= new Random();
int nr = generator.nextInt(valMax));
....
```

Problema 4

Implementați o clasă care să conțină, în metoda `main`, două obiecte de tip `Vector`, reprezentând două mulțimi cu numere întregi. Considerând că nu există elemente duplicate, să se realizeze operațiile elementare cu mulțimi: reuniunea, intersecția și diferența dintre prima mulțime și a doua.

Pentru testare, se vor introduce în vectori minimum zece elemente. Pentru fiecare operație elementară cu mulțimi, se va folosi un obiect de tip `Vector` pentru a se reține rezultatul.

```
boolean ok = v.contains(2);
```

Problema 5

Să se implementeze o clasă, avand ca nume ***Problema5***, care conține, pe lângă metoda ***main***, o metodă care primește două argumente: un șir de caractere constant și un vector de cuvinte (șiruri de caractere).

1. Dacă textul va conține cel puțin o apariție a unui cuvânt din vectorul primit ca parametru, se va afișa mesajul **"Text suspect"**, altfel, afișându-se mesajul **"Nimic suspect"**.
2. Metoda va returna un șir de caractere în care fiecare apariție a unui cuvânt, din vectorul de cuvinte, este cenzurată. De exemplu, aparițiile cuvântului `terorist` se vor înlocui cu `t* * * * * *t`.

```
String text = "Un terorist avea o bomba";
String cuvinte[] = new String[2];
cuvinte[0] = "terorist";
cuvinte[1] = "bomba";
Problema5 prb5 = new Problema5();
String rezultat;
rezultat = prb5.cenzurare(text, cuvinte);
```

Problema 6

Să se realizeze o clasă care să conțina, în metoda `main`, un obiect de tip `Vector` în care să se introducă mai multe tipuri de obiecte (`int`, `double`, `float`, `String`, `char`, `boolean`). Pentru fiecare tip, să se determine câte elemente de acest tip există în vector (se va folosi un al doilea vector în care se va introduce numărul de apariții).

```
Vector v = new Vector();
v.add(7.5);
v.add("String");
System.out.println(v.get(0).getClass());
System.out.println(v.get(1).getClass());
```

poo/laboratoare/02.txt · Last modified: 2018/10/02 22:48 by mihai.nan